```
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
```

```
VV      VV    AAAAAA    XX      XX  77777777    5555555555    000000    RRRRRRRR    EEEEEEEEEE    GGGGGGGG
VV      VV    AAAAAA    XX      XX  77777777    5555555555    000000    RRRRRRRR    EEEEEEEEEE    GGGGGGGG
VV      VV  AA      AA  XX      XX          77  55          00      00  RR      RR  EE          GG
VV      VV  AA      AA  XX      XX          77  55          00      00  RR      RR  EE          GG
VV      VV  AA      AA    XX  XX            77  555555      00    0000  RR      RR  EE          GG
VV      VV  AA      AA    XX  XX            77  555555      00    0000  RR      RR  EE          GG
VV      VV  AA      AA      XX            77          55  00  00  00  RRRRRRRR    EEEEEEEE    GG
VV      VV  AA      AA      XX            77          55  00  00  00  RRRRRRRR    EEEEEEEE    GG
VV      VV  AAAAAAAAAA    XX  XX        77              55  0000    00  RR  RR      EE          GG  GGGGGG
VV      VV  AAAAAAAAAA    XX  XX        77              55  0000    00  RR  RR      EE          GG  GGGGGG
  VV  VV    AA      AA  XX      XX    77        55      55  00      00  RR    RR    EE          GG      GG      ....
  VV  VV    AA      AA  XX      XX    77        55      55  00      00  RR    RR    EE          GG      GG      ....
    VV      AA      AA  XX      XX  77            555555    000000    RR      RR  EEEEEEEEEE    GGGGGG          ....
    VV      AA      AA  XX      XX  77            555555    000000    RR      RR  EEEEEEEEEE    GGGGGG          ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
      SUBROUTINE VAX750REG (LUN,REGISTER)
C
C Version:      'V04-000'
C
C*****************************************************************************
C*                                                                           *
C*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
C*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
C*  ALL RIGHTS RESERVED.                                                     *
C*                                                                           *
C*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
C*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
C*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
C*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
C*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
C*  TRANSFERRED.                                                             *
C*                                                                           *
C*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
C*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
C*  CORPORATION.                                                             *
C*                                                                           *
C*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
C*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
C*                                                                           *
C*                                                                           *
C*****************************************************************************
C
C
C       AUTHOR  BRIAN PORTER            CREATION DATE   16-AUG-1979
C
C       Functional description:
C
C       This module displays the Kernel cluster registers that appear
C       in various error log entries for the 11/750.
C
C       Modified by:
C
C       V03-002 SAR0161         Sharon A. Reynolds,     13-Oct-1983
C               Added an SYE update that fixes carriage control.
C
C       v03-001 BP0001          Brian Porter,           20-AUG-1982
C               Added ci750 support.
C**


        include 'src$:modes.for /nolist'

        BYTE            LUN

        INTEGER*4       REGISTER

        integer*4       lib$extzv

        INTEGER*4       COMPRESSC
```

```
      character*18    v1mcesr(0:0,0:1)

      character*27    v2mcesr(1:3)

      character*17    v1caer(0:3)

      CHARACTER*21    V1CMIER(0:3)

      CHARACTER*7     V2CMIER(4:4)

      character*17    v3cmier(8:11)

      CHARACTER*18    V4CMIER(12:12)

      CHARACTER*9     V5CMIER(18:19,0:1)

      CHARACTER*12    V6CMIER(20:20)



      data    v1mcesr(0,0)    /'OPERAND REFERENCE*'/

      data    v1mcesr(0,1)    /'XB REFERENCE*'/



      data    v2mcesr(1)      /'UNALIGNED UNIBUS REFERENCE*'/

      data    v2mcesr(2)      /'TB PARITY ERROR*'/

      data    v2mcesr(3)      /'BUS ERROR*'/



      data    v1caer(0)       /'CACHE HIT*'/

      data    v1caer(1)       /'2ND ERROR*'/

      data    v1caer(2)       /'DATA PARITY ERROR*'/

      data    v1caer(3)       /'TAG PARITY ERROR*'/



      DATA    V1CMIER(0)      /'CORRECTED READ DATA*'/

      DATA    V1CMIER(1)      /'LOST ERROR*'/

      DATA    V1CMIER(2)      /'READ DATA SUBSTITUTE*'/
```

```
      DATA    V1CMIER(3)      /'MEMORY ERROR*'/


      DATA    V2CMIER(4)      /'TB HIT*'/


      data    v3cmier(8)      /'TB G0 DATA ERROR*'/
      data    v3cmier(9)      /'TB G1 DATA ERROR*'/
      data    v3cmier(10)     /'TB G0 TAG ERROR*'/
      data    v3cmier(11)     /'TB G1 TAG ERROR*'/


      data    v4cmier(12)     /'READ LOCK TIMEOUT*'/


      data    v5cmier(18,0)   /'VIRTUAL*'/
      data    v5cmier(18,1)   /'PHYSICAL*'/
      data    v5cmier(19,0)   /'MODIFY*'/
      data    v5cmier(19,1)   /'READ*'/


      DATA    V6CMIER(20)     /'CMI DISABLE*'/


      character*19    v1cnfgr(8:10)
      data            v1cnfgr(8)      /'POWER-FAIL DISABLE*'/
      data            v1cnfgr(9)      /'TRANSMIT DC LO*'/
      data            v1cnfgr(10)     /'TRANSMIT AC LO*'/
      character*34    v2cnfgr(12:17)
      data            v2cnfgr(12)     /'NO ''CIPA''*'/
      data            v2cnfgr(13)     /'''CIPA'' TIME-OUT*'/
```

```
      data            v2cnfgr(14)     /'DIAGNOSE*'/

      data            v2cnfgr(15)     /'"CIPA" PARITY ERROR*'/

      data            v2cnfgr(16)     /'CORRECTED READ DATA*'/

      data            v2cnfgr(17)     /'UNCORRECTABLE DATA ERROR RESPONSE*'/

      character*29    v3cnfgr(19:20)

      data            v3cnfgr(19)     /'READLOCK TRANSFER TIME-OUT*'/

      data            v3cnfgr(20)     /'NON-EXISTENT MEMORY RESPONSE*'/

      character*19    v4cnfgr(22:23)

      data            v4cnfgr(22)     /'ADAPTER POWER-UP*'/

      data            v4cnfgr(23)     /'ADAPTER POWER-DOWN*'/


      ENTRY COMET_TBDR (LUN,REGISTER)


      CALL LINCHK (LUN,1)

      WRITE(LUN,10) REGISTER
10    FORMAT(' ',T8,'TBGDR',T24,Z8.8)

      if (iand(register,'0000000b'x) .ne. 0) then

      call linchk (lun,1)

      write(lun,15) 'DIAGNOSTIC MODE'
15    format(' ',t40,a)
      endif

      RETURN


      ENTRY COMET_CADR (LUN,REGISTER)


      CALL LINCHK (LUN,1)

      WRITE(LUN,20) REGISTER
20    FORMAT(' ',T8,'CGDR',T24,Z8.8)

      if (iand(register,'00000001'x) .ne. 0) then

      call linchk (lun,1)
```

```
      write(lun,25) 'DIAGNOSTIC MODE'
25    format(' ',t40,a)
      endif

      RETURN



      ENTRY COMET_MCESR (LUN,REGISTER)


      CALL LINCHK (LUN,1)

      WRITE(LUN,30) REGISTER
30    FORMAT(' ',T8,'MCESR',T24,Z8.8)

      call output (lun,register,v1mcesr,0,0,0,'2')

      call output (lun,register,v2mcesr,1,1,3,'0')

      RETURN



      ENTRY COMET_CAER (LUN,REGISTER)


      CALL LINCHK (LUN,1)

      WRITE(LUN,35) REGISTER
35    FORMAT(' ',T8,'CAER',T24,Z8.8)

      CALL OUTPUT (LUN,REGISTER,V1CAER,0,0,3,'0')

      RETURN

C
C     THE CMIER REGISTER IS USED BY SEVERAL MODULES.THE
C     MICRO-CODE WITHIN COMET SPLITS THE REGISTER INTO
C     INDIVIDUAL 4 BIT SLICES AND PUSHES THAT RIGHT
C     JUSTIFIED ONTO THE STACK WHEN A MACHINE CHECK
C     OCCURS.  ANY MODULE CALLING CMIER_XXX MUST
C     PRESENT THE BITS IN THEIR CORRECT POSITION.
C     ALSO THE MODULE MUST HANDLE THE UNUSED BITS
C     WITHIN THE BIT SLICE ITSELF.
C


      ENTRY CMIER_30 (LUN,REGISTER)


```

```
      CALL OUTPUT (LUN,REGISTER,V1CMIER,0,0,3,'0')

      RETURN


      ENTRY CMIER_74 (LUN,REGISTER)


      call output (lun,register,v2cmier,4,4,4,'0')

      RETURN


      ENTRY CMIER_118 (LUN,REGISTER)


      call output (lun,register,v3cmier,8,8,11,'0')

      RETURN


      ENTRY CMIER_1512 (LUN,REGISTER)


      CALL OUTPUT (LUN,REGISTER,V4CMIER,12,12,12,'0')

      RETURN


      ENTRY CMIER_1916 (LUN,REGISTER)



      CALL LINCHK (LUN,1)

      WRITE(LUN,75) modes(lib$extzv(16,2,register))
75    FORMAT(' ',T40,'CPU MODE = ',
     1 A<COMPRESSC (modes(lib$extzv(16,2,register)))>)

      call output (lun,register,v5cmier,18,18,19,'2')

      RETURN


      ENTRY CMIER_2320 (LUN,REGISTER)
```

```
0380
0381            if (iand(register,'00100000'x) .ne. 0) then
0382
0383            call linchk (lun,1)
0384
0385            write(lun,80) 'DIAGNOSTIC MODE'
0386    80      format(' ',t40,a)
0387            endif
0388
0389            RETURN
0390
0391
0392
0393            entry ci750_cnfgr (lun,register)
0394
0395
0396
0397            call linchk (lun,1)
0398
0399            write(lun,85) register
0400    85      format(/' ',t8,'CNFGR',t24,z8.8)
0401
0402            call linchk (lun,1)
0403
0404            if (lib$extzv(0,8,register) .ne. '38'x) then
0405
0406            write(lun,90) 'ADAPTER NOT CI750'
0407    90      format(' ',t40,a)
0408            else
0409
0410            write(lun,90) 'ADAPTER IS CI750'
0411
0412            call output (lun,register,v1cnfgr,8,8,10,'0')
0413
0414            call output (lun,register,v2cnfgr,12,12,17,'0')
0415
0416            call output (lun,register,v3cnfgr,19,19,20,'0')
0417
0418            call output (lun,register,v4cnfgr,22,22,23,'0')
0419            endif
0420
0421            return
0422
0423            end
```

PROGRAM SECTIONS

	Name	Bytes	Attributes
0	$CODE	1079	PIC CON REL LCL SHR EXE RD NOWRT LONG
1	$PDATA	258	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2	$LOCAL	1672	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
3	MODE	55	PIC OVR REL GBL SHR NOEXE RD WRT LONG
	Total Space Allocated	3064	

ENTRY POINTS

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
0-00000306		CI750_CNFGR	0-00000200		CMIER_118	0-00000223		CMIER_1512	0-00000246		CMIER_1916
0-000002BC		CMIER_2320	0-000001BA		CMIER_30	0-000001DD		CMIER_74	0-00000086		COMET_CADR
0-00000166		COMET_CAER	0-000000FD		COMET_MCESR	0-0000000F		COMET_TBDR	0-00000000		VAX750REG

VARIABLES

Address	Type	Name	Address	Type	Name
AP-00000004@	L*1	LUN	2-00000300@	I*4	REGISTER

ARRAYS

Address	Type	Name	Bytes	Dimensions
3-00000000	CHAR	MODES	55	(0:4)
2-00000075	CHAR	V1CAER	68	(0:3)
2-000000B9	CHAR	V1CMIER	84	(0:3)
2-0000019A	CHAR	V1CNFGR	57	(8:10)
2-00000000	CHAR	V1MCESR	36	(0:0, 0:1)
2-0000010D	CHAR	V2CMIER	7	(4:4)
2-000001D3	CHAR	V2CNFGR	204	(12:17)
2-00000024	CHAR	V2MCESR	81	(3)
2-00000114	CHAR	V3CMIER	68	(8:11)
2-0000029F	CHAR	V3CNFGR	58	(19:20)
2-00000158	CHAR	V4CMIER	18	(12:12)
2-000002D9	CHAR	V4CNFGR	38	(22:23)
2-0000016A	CHAR	V5CMIER	36	(18:19, 0:1)
2-0000018E	CHAR	V6CMIER	12	(20:20)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
1-00000074	10'	1-00000086	15'	1-0000008D	20'	1-0000009E	25'	1-000000A5	30'	1-000000B7	35'
1-000000C8	75'	1-000000E1	80'	1-000000E8	85'	1-000000FB	90'				

FUNCTIONS AND SUBROUTINES REFERENCED

Type	Name	Type	Name	Type	Name	Type	Name
I*4	COMPRESSC	I*4	LIB$EXTZV		LINCHK		OUTPUT

COMMAND QUALIFIERS

```
FORTRAN /LIS=LIS$:VAX750REG/OBJ=OBJ$:VAX750REG MSRC$:VAX750REG

/CHECK=(NOBOUNDS,OVERFLOW,NOUNDERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/STANDARD=(NOSYNTAX,NOSOURCE_FORM)
/SHOW=(NOPREPROCESSOR,NOINCLUDE,MAP)
/F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOCROSS_REFERENCE  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
Run Time:            3.78 seconds
Elapsed Time:        11.36 seconds
Page Faults:         164
Dynamic Memory:      187 pages
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

EVLDEF
MDL

LIBTAIL
B32

XXXQUE
LIS

UTILDEF
REQ

ERRDEF
MDL

EVCDEF
MDL

CONSOLE
LIS

USEROPEN
LIS

VAXPSL
LIS

ERRFMT

VAX780REG
LIS

ERRFMT
MAP

EVLIBRARY
B32

VAX750REG
LIS

VECMAPREG
LIS

ERRFMT
LIS

EVL

LIBHEAD
B32

EVL
MAP

WQDEF
MDL